Sessione ordinaria 2009
Seconda prova scritta

M070 – ESAME DI STATO DI ISTITUTO TECNICO INDUSTRIALE

CORSO DI ORDINAMENTO

Indirizzo: INFORMATICA

Tema di: INFORMATICA
(Testo valevole per i corsi di ordinamento e per i corsi sperimentali del Progetto "Sirio" – Informatica)

Una società telefonica desidera dotarsi di un sistema informativo che consenta ai propri tecnici l'accesso on line ad una rubrica anagrafica dei contatti, in modo che ciascun componente del gruppo possa consultare in ogni momento le informazioni in essa contenute e all'occorrenza aggiornarle (inserire nuovi contatti, modificare o eliminare contatti esistenti).

L'Amministratore del sistema informativo dovrà ampliare il portale al quale accedono abitualmente i membri del gruppo aggiungendo questa nuova funzione: ogni componente dovrà essere autenticato dal sistema in uso mediante inserimento di credenziali (username e password) in modo da garantire un accesso sicuro alle informazioni per cui è autorizzato; le informazioni sono condivise tra più utenti del gruppo di lavoro.

Al fine di produrre periodicamente statistiche per ottimizzare il lavoro del gruppo si deve tenere traccia di ogni accesso al sistema, registrandone le informazioni essenziali (identificatore utente, data e ora di accesso, data e ora di fine sessione, …) e le operazioni (consultazioni/aggiornamenti della rubrica dei contatti) svolte da ciascun utente autenticato; nel caso di operazioni di aggiornamento è opportuno mantenere anche lo stato della rubrica prima della variazione fino a che l'Amministratore del sistema validerà le modifiche e le renderà pubbliche.

Il candidato, fatte le opportune ipotesi aggiuntive:

1. produca un'analisi della realtà di riferimento ponendo attenzione sugli aspetti riguardanti la sicurezza del sistema informativo
2. progetti uno schema concettuale e il corrispondente schema logico del data base
3. realizzi la definizione delle relazioni in linguaggio SQL e le seguenti interrogazioni espresse in linguaggio SQL:
   - visualizzare l'elenco, in ordine alfabetico per denominazione, dei contatti di una determinata provincia
   - elencare, in ordine temporale, gli accessi effettuati da un membro del gruppo
   - calcolare e visualizzare il numero medio giornaliero di accessi in un determinato periodo di tempo
   - calcolare e visualizzare il numero totale di nuovi contatti inseriti, per ogni componente del gruppo
   - elencare le operazioni effettuate in un determinato giorno da ogni utente del gruppo di lavoro
   - visualizzare le informazioni del contatto più consultato nell'arco di una settimana
4. proponga una soluzione per l'amministrazione via web del sistema e codifichi in un linguaggio di programmazione a scelta un segmento significativo del progetto realizzato.

Durata massima della prova: 6 ore.
È consentito soltanto l'uso di manuali tecnici e di calcolatrici non programmabili.
Non è consentito lasciare l'Istituto prima che siano trascorse 3 ore dalla dettatura del tema.

Una soluzione «semplificata»

Come avviene ormai da diversi anni la traccia ministeriale richiede l'analisi e la progettazione di un database molto articolato che deve essere gestito, per la consultazione e l'aggiornamento, sia in ambiente locale (la sede delle compagnia telefonica), sia via Internet. Le richieste della traccia sono molto dettagliate e articolate e richiederebbero per una soluzione completa da parte dello studente un tempo molto maggiore di quello a disposizione. Sicuramente il tema proposto, pur trattando argomenti che dovrebbero essere noti al candidato, sempre più adatto ad un progetto di ampio respiro da sviluppare in classe in un periodo di tempo piuttosto lungo che non per un tema di maturità. Infatti le richieste spaziano da un'analisi approfondita delle tematiche legate alla sicurezza e alla riservatezza delle informazioni, ad un progetto completo di database in cui le modalità di salvaguardia delle informazioni nel tempo sono lasciate a carico del candidato. Inoltre vengono richieste diverse query che richiedono la conoscenza approfondita delle possibilità offerte dal linguaggio SQL. Il punto numero 4, infine, allarga l'analisi alle tematiche relative alla gestione di un database remoto e alla realizzazione in Internet di un portale interattivo con accessi riservati, argomento che probabilmente da solo potrebbe rappresentare una parte cospicua della soluzione. Per questo motivo proporrò una soluzione "semplificata" che penso possa essere compatibile con il tempo a disposizione.

Da un punto di vista generale il problema risulta essere molto articolato in quanto richiederebbe un'analisi approfondita delle tecniche da utilizzare per garantire da una parte la sicurezza e la coerenza dei dati nel tempo, dall'altra la riservatezza delle informazioni che dovrebbero essere accessibili solo alle persone autorizzate. In particolare voglo accennare al fatto che, essendoci nel database informazioni riservate come le password, sarebbe opportuno registrare nel database dei dati crittografati definendo nel contempo l'algoritmo di codifica e decodifica. Inoltre occorre distinguere tra le operazioni di competenza dell'Amministratore (aggiunta e modifica dei dati relativi ai tecnici, assegnazione di userid e password iniziale, validazione delle modifiche ai dati dei contatti, …) e quelle di competenza dei tecnici (modifica della propria password, consultazione e modifica dei dati relativi ai propri contatti, consultazione e/o modifica dei dati dei contatti degli altri componenti del proprio gruppo, …) Infine occorre tener presente che nel lasso di tempo intercorrente tra l'accesso da parte di un tecnico per la modifica dei dati e la validazione della modifica stessa e della sua pubblicazione è indispensabile garantire l'integrità e la congruenza dei dati del database (un secondo tecnico non essendo a conoscenza delle modifiche apportate dal collega potrebbe tentare di apportarne altre che potrebbero risultare incompatibili con quelle del collega, dando origine ad un database inconsistente) Tutto ciò richiederebbe la gestione di diverse tabelle con accessi separati e dei permessi per i singoli componenti.

Per semplicità nella nostra soluzione ci atterremo alle seguenti ipotesi restrittive:

- Il database è relativo ad un unico gruppo di lavoro, per cui i tecnici hanno libero accesso a tutti i contatti;
- Non vengono prese in esame le problematiche legate alla crittografia dei dati e alla gestione delle password;
- Il sistema una volta accettata una modifica sui dati di un contatto ne impedisce l'accesso per modifica o cancellazione fino alla validazione da parte dell'Amministratore.

SVOLGIMENTO 2008/2009

PREMESSE:

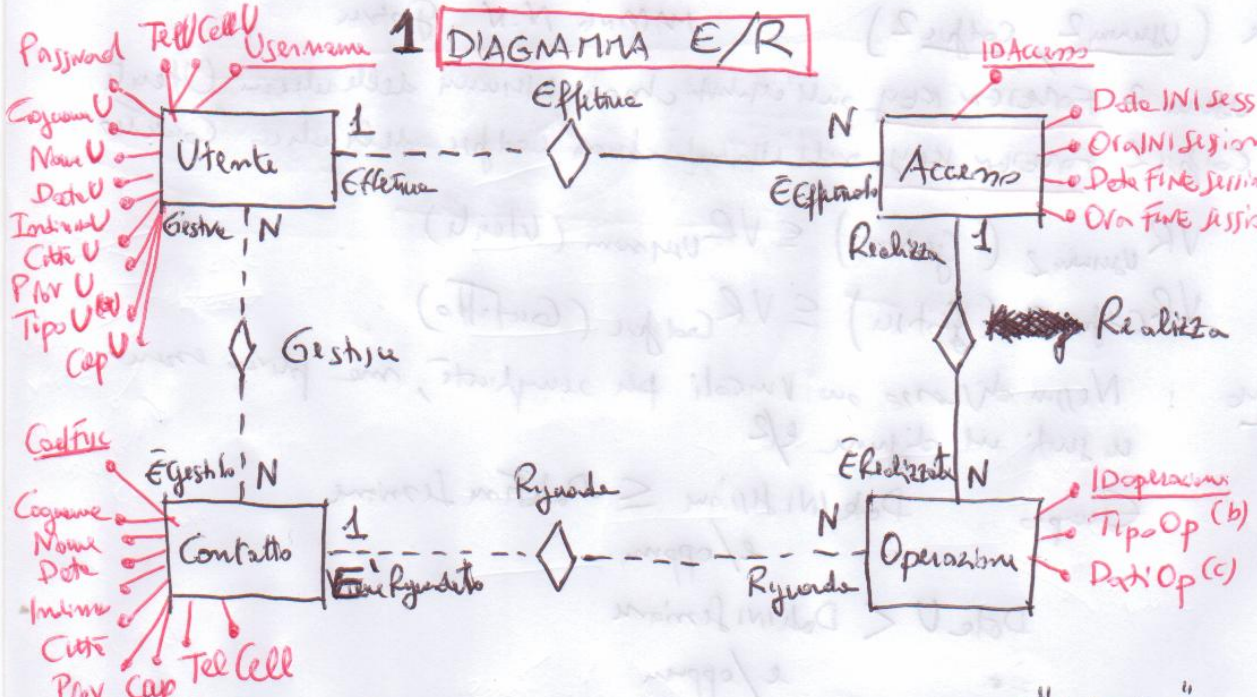
① IL TESTO DISTINGUE TRA (RIGA 2 e 3 DEL TESTO)
- CONSULTARE ≡ QUERY
- AGGIORNARE { INSERT, UPDATE, DELETE } LE INFORMAZIONI

② IL ROGALE SI IBRIDIZZA GLI ESISTENTE ED UTILIZZATO DA ALTRI MEMBRI DEL GRUPPO (NON PER FORZA TECNICI) E DALLO STESSO AMMINISTRATORE DEL SISTEMA INFORMATIVO (RIGA 5 e 6 DEL TESTO)

③ IN CASO DI OPERAZIONI DI AGGIORNAMENTO (OSSIA INSERT, UPDATE E DELETE PER QUANTO DETTO AL PUNTO ①) È OPPORTUNO MANTENERE LO STATO DELLA RUBRICA PRIMA DELLA VARIAZIONE FINO A CHE L'AMMINISTRATORE NON LE VALIDI (RIGA 12-13-14 DEL TESTO) E LE RENDA PUBBLICHE.

④ NELLA SOLUZIONE PROPOSTA SI IBRIDIZZANO OPERAZIONI DI CONSULTAZIONE (query) ED AGGIORNAMENTI (INSERT, UPDATE E DELETE) PUNTUALI OSSIA RIGUARDANTI AL PIÙ UN RECORD ALLA VOLTA.

1 DIAGRAMMA E/R



NOTA BENE:

(a) L'attributo Tipo U dell'entità "Utente" può valere solo { "Admin", "Tecnico", "Amministrativo" }

(b) L'attributo TipoOp dell'entità "Operazione" può valere solo { "QUERY", "INSERT", "UPDATE", "DELETE" }

(c) L'attributo DatiOp dell'entità "Operazione" nel caso che TipoOp valga "INSERT", "UPDATE" o "DELETE" conterrà una stringa di testo così costruita

<CodFisc>#<Cognome>#<Nome>#<Data>#<Indirizzo>#<Città>#<Prov>

PER POTER RISPONDERE A QUANTO RICHIESTO AL PUNTO ③

2 SCHEMA RELAZIONALE

(•) Utente (Username, Password, CognomeU, NomeU, DataU, IndirizzoU, CittàU, ProvU, TipoU, CapU, TelU, CellU)

(•) Accesso (IDAccesso, DataINIsessione, OraINIsessione, DataFINEsessione, OraFINEsessione, Username1)

con UserNome1 FOREIGN KEY sull'attributo chiave Username della relazione relazione Utente

OSSIA $VR_{Username1}(Accesso) \subseteq VR_{Username}(Utente)$

(•) Operazione (IDoperazione, TipoOp, DataOp, IDAccesso1, codfisc1)

con IDAccesso1 FOREIGN KEY sull'attributo chiave IDAccesso della relazione Accesso

con codfisc1 FOREIGN KEY sull'attributo chiave Codfisc della relazione Cantiere

OSSIA $VR_{IDAccesso1}(Operazione) \subseteq VR_{IDAccesso}(Accesso)$ TOTALITÀ ASS. INVERSA

" " $\supseteq$ " " TOTALITÀ ASS DIRETTA

(•) Cantiere (Codfisc, Cognome, Nome, Data, Indirizzo, Città, Prov, Cap, Tel, Cell)

(•) Gestisce (Username2, Codfisc2) MAPPING N:N Gestisce

con Username2 FOREIGN KEY sull'attributo chiave Username della relazione Utente

con Codfisc2 FOREIGN KEY sull'attributo chiave Codfisc della relazione Cantiere

OSSIA $VR_{Username2}(Gestisce) \subseteq VR_{Username}(Utente)$

$VR_{Codfisc2}(Gestisce) \subseteq VR_{Codfisc}(Cantiere)$

NOTA BENE : Nessun discorso sui vincoli per semplicità, ma potrebbero essere inseriti nel diagramma E/R

Esempio DataINIsessione $\leq$ DataFineSessione

e/oppure

DataU $<$ DataINIsessione

e/oppure

Data $<$ DataINIsessione

3 DEFINIZIONE RELAZIONI IN SQL

```
CREATE DATABASE miodb;

CREATE DOMAIN operazionedb AS char(6) CHECK
    (operazionedb IN ("QUERY","INSERT","UPDATE","DELETE"));
CREATE DOMAIN tipoutente AS char(10) CHECK (tipoutente IN ("Admin","Utente",
                                                "Segretario"));
CREATE TABLE Utente
  (
    Username  CHAR(10)   NOT NULL,
    Password  Char(10)   NOT NULL,
    CognomeU  char(10)   NOT NULL,
    NomeU     Char(10)   NOT NULL,
    DataU     char(10)   NOT NULL
    IndirizzoU char(100) NOT NULL,
    CittàU    Char(20)   NOT NULL,
    PrvU      char(2)    NOT NULL,
    TipoU     tipoutente NOT NULL,
    CapU      char(5)    NOT NULL,
    TelU      char(20)   NOT NULL,
    CellU     char(20)   NOT NULL,
    PRIMARY KEY (Username),
    UNIQUE (CognomeU, NomeU, DataU),
  );
CREATE TABLE Accesso
  (IDAccesso  Char(10)  NOT NULL,
   DataINIsessione Date  NOT NULL,
   OraINIsessione Time  NOT NULL,
   DataFINEsessione Data NOT NULL,
   OraFINEsessione Time NOT NULL,
   Username1  Char(10) NOT NULL,
   PRIMARY KEY (IDAccesso),
   FOREIGN KEY (Username1) REFERENCES utenti (Username) ON DELETE CASCADE
   [Check (DataINIsessione ≤ DataFINEsessione)]
  );
```

```
CREATE TABLE Operazione
(
  IDoperazione   Char(10)   NOT NULL,
  Tipo Op        operazione_db NOT NULL,
  Dok op         char(200)
  IDAccount 1    char(10)   NOT NULL,
  Codfisc 1      char(16)   NOT NULL,
  PRIMARY KEY (IDoperazione),
  FOREIGN KEY (IDAccount 1) REFERENCES Account(IDAccount) ON DELETE CASCADE,
  FOREIGN KEY (Codfisc1) REFERENCES Conteto(Codfisc) ON DELETE CASCADE
);

CREATE TABLE Conteto
(
  Codfisc   Char(16)   NOT NULL,
  Cognome   Char(30)   NOT NULL,
  Nome      Char(30)   NOT NULL,
  Date      date       NOT NULL,
  Indirizzo Char(100)  NOT NULL,
  Cap       Char(20)   NOT NULL,
  Prov      Char(2)    NOT NULL,
  Cap       Char(5)    NOT NULL,
  Tel       char(20)   NOT NULL,
  Cell      Char(20)   NOT NULL,
  PRIMARY key (Codfisc),
);                      UNIQUE (Cognome, Nome, Date)

CREATE TABLE Gestisce
(
  Username2   char(10)  NOT NULL,
  Codfisc2    char(16)  NOT NULL,
  PRIMARY KEY (Username2, Codfisc2),
  FOREIGN KEY (Username2) REFERENCES Utente(Username) ON DELETE CASCADE,
  FOREIGN KEY (Codfisc2) REFERENCES Conteto(Codfisc) ON DELETE CASCADE
);
```

4 SVOLGIMENTO QUERY

Q1: VISUALIZZARE L'ELENCO, IN ORDINE ALFABETICO PER DENOMINAZIONE, DEI CONTATTI DI UNA DETERMINATA PROVINCIA

```
Q1 = SELECT *
     FROM Contatto
     WHERE Prov = [Prov_Immessa]
     ORDER BY Cognome [ASC], Nome [ASC] ;
```

Q2: ELENCARE, IN ORDINE TEMPORALE, GLI ACCESSI EFFETTUATI DA UN MEMBRO DEL GRUPPO

```
Q2 = SELECT *
     FROM Accesso
     WHERE Username1 = [Username_membro]
     ORDER BY DataINI Sessione [ASC], OraINI Sessione [ASC] ;
```

Q3: CALCOLARE E VISUALIZZARE IL NUMERO MEDIO GIORNALIERO DI ACCESSI IN UN DETERMINATO PERIODO DI TEMPO

```
Q3 = SELECT AVG (Temptab.TotAccessi) AS MediaAccessi
     FROM (SELECT COUNT(*) AS TotAccessi
           FROM Accesso
           WHERE DataINI Sessione ≥ [DataINI_Immessa] AND
                 DataFINE Sessione ≤ [DataFINE_Immessa]
           GROUP BY DataINI Sessione) AS Temptot ;
```

Q4: CALCOLARE E VISUALIZZARE IL NUMERO TOTALE DI NUOVI CONTATTI INSERITI, PER OGNI COMPONENTE DEL GRUPPO

```
Q4 = SELECT CognomeU, NomeU, COUNT(*) AS TotContatti
     FROM Operazione, Accesso, Utente
     WHERE (IDAccesso1 = IDAccesso) AND (Username1 = Username)
           AND TipoOp = "INSERT"
     GROUP BY CognomeU, NomeU
     HAVING TipoU = "Tecnico" ;
```

Q5: ELENCARE LE OPERAZIONI EFFETTUATE IN UN DETERMINATO GIORNO DA OGNI UTENTE DEL GRUPPO DI LAVORO

```
Q5 = SELECT CognomeU, NomeU, TipoOp
     FROM Operazione, Accesso, Utente
     WHERE (IDAccesso1 = IDAccesso) AND (Username1 = Username)
           AND TipoUtente = "Tecnico"
           AND DataINI Sessione = [DataRiferimento_Immessa] ;
```

Q_6: VISUALIZZARE LE INFORMAZIONI DEL CONTATTO PIÙ CONSULTATO NELL'ARCO DI UNA SETTIMANA

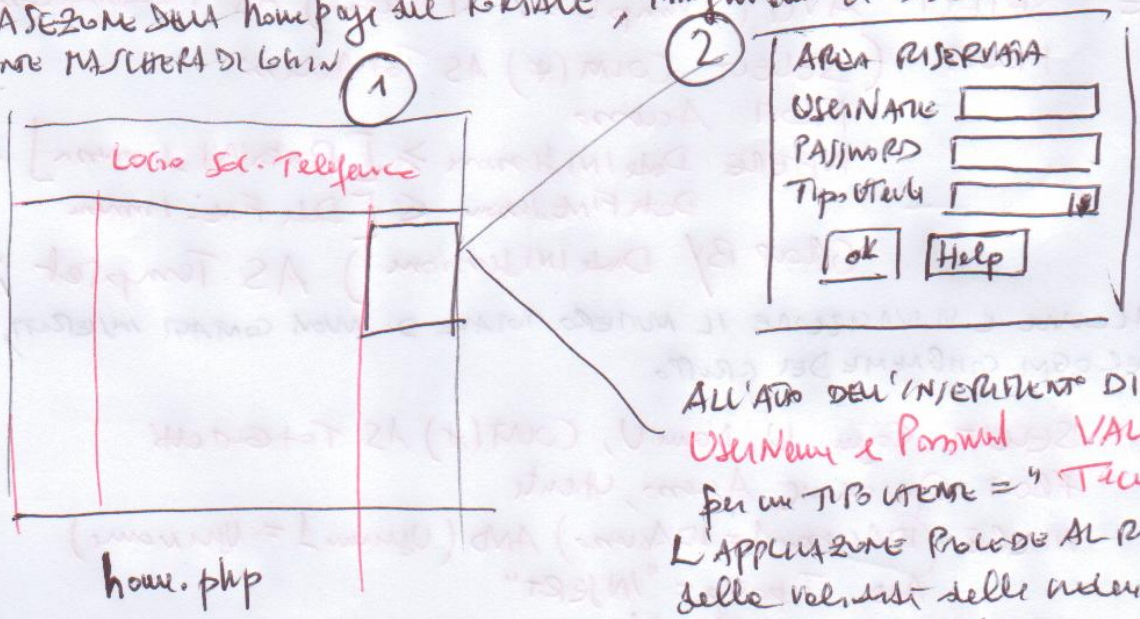
```
Q6 = SELECT MAX (TempTab.TotAccessi) AS MaxAccessi
     FROM ( SELECT COUNT(*) AS TotAccessi
            FROM Utenti, Accessi, Operazioni, Contatti
            WHERE (Utenti.Username1 = Username) AND (IDAccessi1 = IDAccesso)
                  AND (CodFisc1 = CodFisc) AND TipoOp = "QUERY"
                  AND (DataINSessione >= [DataINInserimento]
                  AND DataFineSessione <= DataINSessione + 7)
            GROUP BY CodFisc ) AS TempTab;
```

N.B. La query presentata fornisce solo il numero MAX di accessi alla tupla considerazione effettuata.

5 IMPLEMENTAZIONE PHP-MySQL

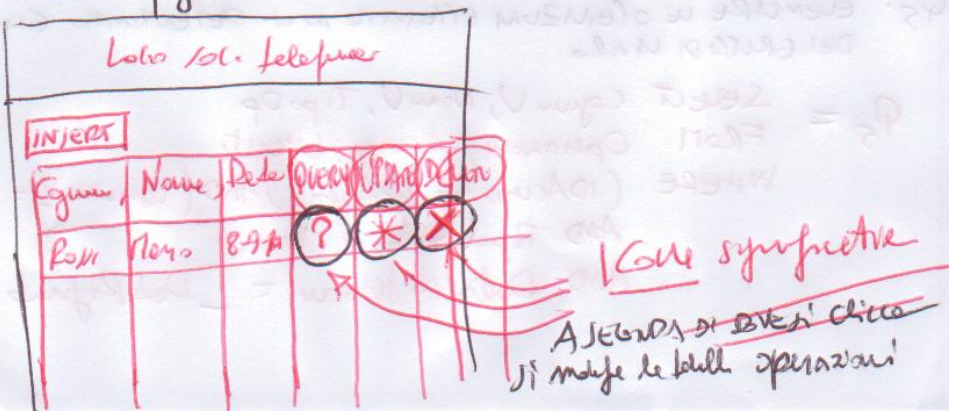
MASCHERA DI LOGIN DELL'APPLICAZIONE

IN UNA SEZIONE DELLA homepage del PORTALE, immaginiamo di DISEGNARE LA SEGUENTE MASCHERA DI LOGIN



ALL'ATTO DELL'INSERIMENTO DI UserName e Password VALIDE per un TIPOUTENTE = "Tecnico" L'APPLICAZIONE PROCEDE AL RICONOSCIMENTO della validità delle credenziali ed all'INSERT dei dati di Accesso nella tabella Accessi.

(3) INDIRIZZO OK si accede al mapping



Questo punto richiederebbe uno sviluppo molto articolato che però a mio avviso va al di là delle possibilità fornite al candidato dal tempo a disposizione. Mi limiterò quindi ad indicare dei criteri di carattere generale.
Dato che viene richiesto la gestione dei contatti dati attraverso un portale, il database deve essere disponibile su un server on line. Per far ciò si può pensare di utilizzare uno dei tanti tool presenti sul mercato per generare la parte grafica e uno dei tanti linguaggi attualmente esistente per la realizzazione di pagine dinamiche per il codice.
Molte sono le opzioni disponibili per lo sviluppo della componente server-side. Nella soluzione che presentiamo, abbiamo deciso di utilizzare il linguaggio di programmazione PHP che consente di arricchire le pagine Web di codice script che sarà eseguito direttamente sul server.
In particolare, PHP consente di implementare un motore di scripting server side molto diffuso e multipiattaforma con un buon supporto della connettività verso database diversi (ad es. dBase, Oracle, MySQL) attraverso componenti standard.
Nello specifico ipotizzeremo di interfacciarsi verso un DBMS MySQL, anch'esso multipiattaforma e piuttosto semplice da utilizzare
Ovviamente, per potere utilizzare PHP è necessario aver installato sul proprio sistema un Web Server (come ad esempio Apache).
Riepilogando si è deciso di utilizzare quindi per la realizzazione del portale e del relativo servizo Web una piattaforma **WAMP** basata su
Sistema Operativo.: **W**indows 2003 server
Web Server: **A**PACHE
Database Server: **M**YSQL
Linguaggio di programmazione lato server: **P**HP
mentre si utilizzerà il **linguaggio SQL** per la creazione delle tabelle e per le interrogazioni.
Esistono piattaforme complete open-source assolutamente gratuite che contengono il web server APACHE, il linguaggio di scripting lato server PHP ed il database relazionale MYSQL, (Ad esempio EasyPHP, PHPMyAdmin oppure Apache2triad) sia per ambiente Windows sia per ambiente Linux.
Naturalmente l'uso di una piattaforma **LAMP** (ossia con un operativo server anch'esso open source come **L**inux) sebbene da un lato porti l'indiscutibile vantaggio della totale gratuità rispetto agli alti costi di gestione della Microsoft, dall'altro può creare a tutti quegli utenti "non addetti ai lavori" che decidano di farne uso e che non abbiano un grosso livello di competenza tecnica informatica specifica più di qualche problemino di configurazione, di customizzazione e di utilizzo non sempre compatibili con le ragioni di efficienza aziendale
Infine va notato che per garantire un discreto margine di sicurezza, sarebbe opportuno far sì che le informazioni riservate, come ad esempio le password, non compaiano nel sorgente HTML della pagina, e che inoltre, tramite opportuni meccanismi, viaggino criptate attraverso la rete.

Immaginiamo che il portale già esista ed abbia come pagina iniziale il file **home.php**
Tale file ospita in un'apposita sezione una maschera di LOGIN all'area riservata che si effettua digitando negli appositi campi USERNAME, PASSWORD e scegliendo da una LISTBOX la voce "Tecnico. La pressione del tasto OK invoca il file **contatti.php** che si occupa di controllare se l'utente è autorizzato e nel caso lo sia ad inserire un record, con i campi opportunamente valorizzati, nella tabella "Accesso"
La pagina **contatti.php,** se l'utente è un tecnico autorizzato**,** gli permette di effettuare una qualsiasi delle operazioni possibili (consultazione ossia QUERY e/o aggiornamenti ossia INSERT, UPDATE o DELETE)

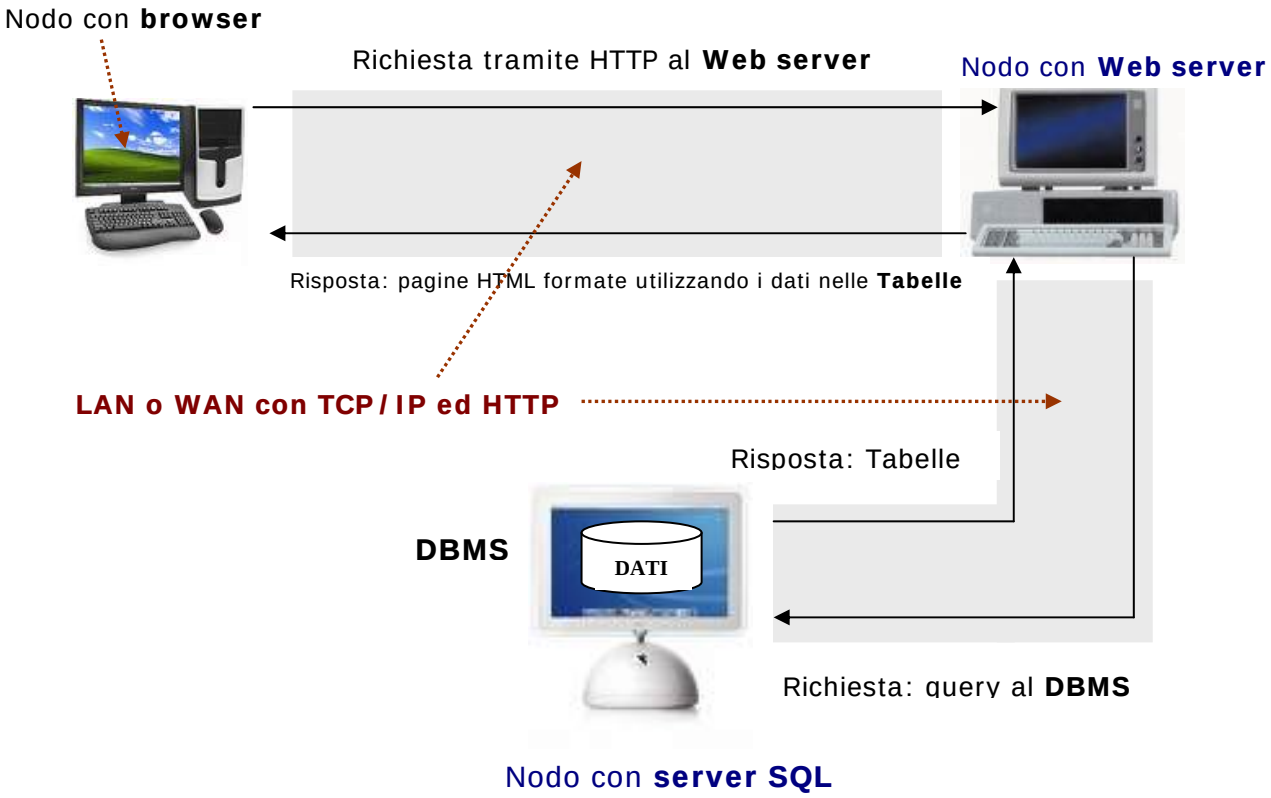
SCHEMA DI MASSIMA file contatti.php (senza dettagli HTML embedded)

```
<? php //Segnalatore di inzio codice PHP
/*****************************************/
/* Valorizzazione parametri del database */
/*****************************************/
$db_host = "62.154.198.216";   // IP ADDRESS del database server che ospita i dati
$db_user = "root";
$db_pswd = "1234";  //per semplicità la pswd è in chiaro e non crittografata
$db_name = "miodb";
/*****************************************/
/* Connessione al database */
/*****************************************/
$connessione = mysql_connect($db_host, $db_user, $db_pswd)
or die("Connessione non riuscita" . mysql_error());
/*****************************************/
/* Selezione del database */
/*****************************************/
mysql_select_db($db_name) or die("Selezione del database non riuscita" . mysql_error());
/*****************************************/
/* Preparazione della query SQL */
/*****************************************/
//occorre costruire nella variabile $query, secondo la sintassi SQL, la query di interesse che controlla se
// l'utente è un tecnico autorizzato all'ingresso
//SELECT *
//FROM Utente
//WHERE ( (Username = $_POST["USERNAME"]) AND (Password =$_POST["PASSWORD"]) AND
//(TipoU=$_POST["TIPOUTENTE"] ) )
/*****************************************/
/* Esecuzione di una query SQL */
/*****************************************/
$risultato = mysql_query($query) or die("Query fallita" . mysql_error());
/*****************************************/
/* Calcolo del numero di occorrenze */
/*****************************************/
$numrighe=mysql_num_rows($risultato);
/*****************************************************************************************************/
/* Visualizzazione delle occorrenze in una pagina HTML usando i relativi TAG in modo embedded (funzione "echo")*/
/*****************************************************************************************************/
if ($numrighe = = 1)
   {
  /*****************************************/
  /* Ciclo sul numero di record trovati dalla query */
  /*****************************************/
   for ($i=0;$i<$numrighe;$i++)
      {
       // Esecuzione di un'altra query che prelevi tutti i record trovati nella tabella Contatti presenti nel db e che
       // costruisca una opportuna tabella di riepilogo (usando i TAG HTML "table")  che permetta di effettuare tutte
       // le operazioni previste
       }
    }
else
   {
   echo "CREDENZIALI UTENTE NON RICONOSCIUTE<br>";
   }
/*********************************************/
/* Liberazione delle risorse del risultato della query */
/*********************************************/
mysql_free_result($risultato) or die("Liberazione risorse fallita" . mysql_error());
/*****************************************/
/* Chiusura della connessione al database */
/*****************************************/
mysql_close($connessione) or die("Chiusura connessione fallita" . mysql_error());
?> //Segnalatore di fine codice PHP
```

In questo script abbiamo utilizzato la funzione che riceve come parametro il nome dell'host, il nome dell'utente e la password.
$connessione =* ***mysql_connect****($host, $ user, $ pswd)
per effettuare il collegamento con un host su cui gira un'istanza di MySQL.
Questa funzione restituisce un oggetto che rappresenta integralmente il nostro host e con il quale è possibile utilizzare la funzione
mysql_select_db*($database,[$connessione])*
per selezionare l'istanza corretta del database (essenziale in quanto l'istanza del database server può contenere più di un database).
Dopo aver selezionato il database, è possibile effettuare query su qualsiasi tabella tramite le funzione
$risultato =* ***mysql_query****($query, [$connessione]);
A questo punto l'oggetto *$risultato* contiene il risultato della query.
Per l'estrazione dei dati dalla select abbiamo utilizzato la funzione
mysql_result*($risultato, $i, $nomecampo)*
che restituisce i contenuti di una cella da un risultato MySQL dove l'argomento campo può essere l'indice della riga contenente tutti i dati. Per ottenre solo numeri di riga validi si è utilizzata la funzione
mysql_num_rows ($risultato)
che restituisce il numero di righe in un risultato. Questo comando è valido solo per le istruzioni SELECT.
Questa query ha estratto tutti i record della tabella Utente e con le opportune istruzioni HTML li abbiamo visualizzati all'interno di una tabella.
Per terminare correttamente una sessione di interrogazione ad MYSQL dobbiamo utilizzare le funzioni
mysql_free_result ($risultato);
mysql_clos e($connessione);
che rispettivamente libera tutta la memoria associata all'identificativo del risultato $*risultato e* chiude la connessione al server MySQL associata all'identificativo di connessione $connessione specificato permettendo eventualmente ad un altro client di utilizzarla.

Per quanto riguarda l'architettura di riferimento da utilizzare per accedere un database in rete per questa soluzione implementativa facciamo le seguente ipotesi di partenza:

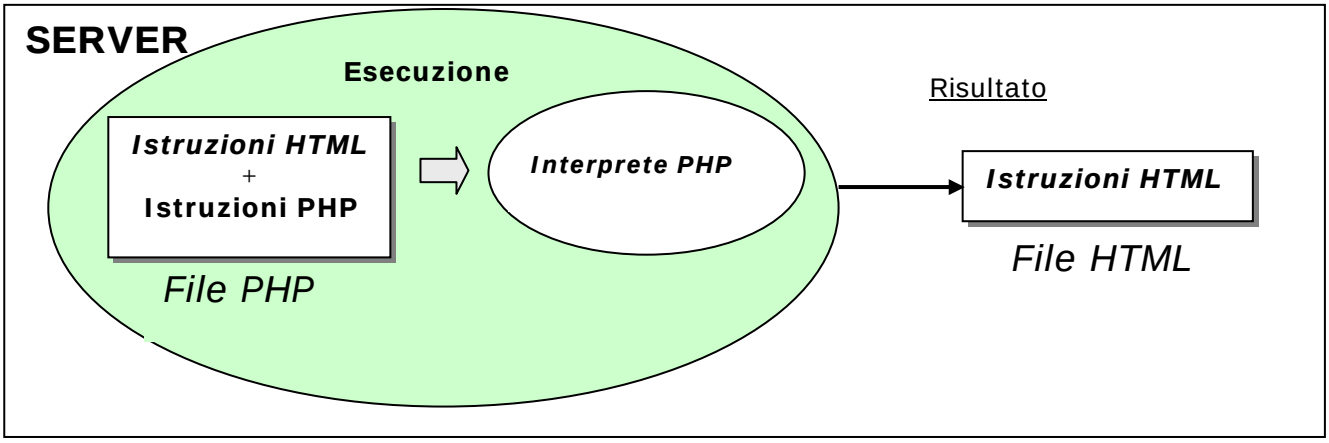
1) consideriamo il modello concettuale di **networking** (ovvero tutto ciò che riguarda la comunicazione tra reti diverse) semplificato a 4 livelli;
2) consideriamo i **protocolli** di comunicazione della famiglia **TCP/IP**, il **browser** come client universale ed il **Web server** come server universale



Per semplicità implementativa, in caso di una quantità di dati non eccessivamente numerosa, possiamo implementare sia il Web server (ad esempio APACHE) sia il Database server (ad esempio MYSQL) **sullo stesso nodo**.
Sempre per semplicità e chiarezza possiamo fare in modo di utilizzare l'**SQL** come linguaggio standard di interrogazione *e* manipolazione di un database ed utilizzeremo anche un **server SQL relazionale** (ad esempio MYSQL).
Dato l'**ambiente client-server** così ipotizzato in un'architettura di **protocolli TCP/IP ed HTTP** possiamo utilizzare per sviluppare tutte le funzionalità previste dall'applicazione complessiva, la **programmazione lato server** ossia possiamo sviluppare una serie di programmi applicativi che andranno in esecuzione prevalentemente sul *server*, accettando le richieste dal client e fornendo a quest'ultimo i risultati dell'elaborazione sottoforma di pagine HTML;

Tra i linguaggi di programmazione lato server possiamo utilizzareil PHP ossia un **linguaggio di scripting lato server** (come **PERL** ed **ASP).** Un linguaggio di scripting si differenzia dai *linguaggi di programmazione veri e propri lato server* perché non ha vita *autonoma* ma può essere utilizzato *esclusivamente* in quel contesto.

Il PHP è **u**n **linguaggio interpretato lato server:** infatti un **file di comandi PHP** è un classico esempio di programma interpretato lato server poiché infatti il Web server associa a tale file l'interprete PHP che deve essere mandato in esecuzione per poterne eseguire i comandi.



Per questo motivo è possibile avere *comandi PHP all'interno di pagine HTML* oppure *pagine PHP pure*.